

APPENDIX

MOH MANAGED OBJECTS AND COMMON OS LAYER MAPPING

MAPPING OF MANAGEMENT LAYER ATTRIBUTES TO THE UNDERLYING LAYERS

Purpose :

This document is targeted towards both the management layer implementors and the common OS layer implementors.

For the management layer implementors it's very important to know how the management layer attribute maps to the class and properties of the common OS layer.

Also it makes an attempt to document the dependencies of the common OS layer properties to the underlying platform and OS. This information is what the common OS layer implementors would like to know.

1.3 PlugInUnit

Moh Class : PlugInUnit

COSL Class : PlugInUnit

<i>Moh Attribute Name</i>	<i>COSL property(s)</i>	<i>Description</i>
Managed Entity Id	id, model, plugInUnitType	In Moh this is the object name or Mbean name of a managed object instance. The COSL properties mentioned in the last column are to be used to form the distinguished name part, the simple name part and the id part. The plugInUnitType and relativeLocation value of the parent slot can be used to form the 'distinguished name' part of the object name, and model & id can be used to form the simple-name and id part of the Mbean name. See 1.5.2 for more on Mbean name.
PlugInUnit Type	plugInUnitType	Since a PlugInUnit object is always the only child of an EquipmentHolder object , the relative location value can be either 0 or 1, ('1' is preferred according to the definition of 'parentRelPos' in ENTITY-MIB). So there is no need for a support from COSL for relativeLocation of Plugins. The value of model can be retrieved from the FRUID information.
		For SUN FRUs, SUN FRUID info can be used to retrieve the plugInUnitType value. For 3 rd party FRUs PICMIG FRUID can be used instead. We need to find out what are the possible plugInUnitType values in phase-1. As an option, for R2.1(release 2 phase 1) we can retain the R1 'plugInUnitType' values. TBD.
		The plugInUnitType value should be "cpcl_card" for an unidentified CPCL card, and that of an unidentified rmm should be "rmm".
Administrative State	NSR	
Availability Status	plugInUnitState	See 1.3.1 and 1.3.1.1
Operational State	plugInUnitState	See 1.3.1 and 1.3.1.1
Vendor Name	vendorName	From FRUID info
Version	version	From FRUID Info
Alarm Severity APR	NSR	

Notification Table :

<i>Moh Notification</i>	<i>SNMP Trap</i>	<i>COSL Event</i>	<i>Description</i>
ObjectDeletionNotification	netraCtHwUnitDeleted	ems_node_removed	For a PlugInUnit object removal from the system.No event is required from COSL for the removal of all the devices under the removed plugin.
ObjectCreationNotification	netraCtHwUnitCreated	ems_node_added	For a PlugInUnit object insertion into the system.

<i>Moh Notification</i>	<i>SNMP Trap</i>	<i>COSL Event</i>	<i>Description</i>
StateChangeNotification	netraCtHwUnitDown/n etraCtHwUnitUp	ems_node_updated	For a change in 'pluginUnitState' property , COSL should notify the agent with this event. But this will depend on the mapping to be defined in section 1.3.1.1.

1.3.1 'pluginUnitState' property :

The following states are defined for 'pluginUnitState' of PluginUnit class at COSL :

'POWEROFF1' : The plugin is **powered off**, also it's **identified**. A newly inserted plugin will be in this state if it supports **FRUID** information. This state is reachable by a CPC1 card plugin supporting HA-hotswap capability.

'POWEROFF2' : The plugin is **powered off**, and **unidentified**. A newly inserted plugin with FRUID support will transit(4a) to 'POWEROFF1' state after the plugin gets identified through the FRUID information.

'UNAVAILABLE1' : The plugin is **powered on, initialized, and identified**. Also it may be electrically connected to the related bus, may be it's software-configured, but it's still not available for use by OS and applications because the necessary drivers are not loaded/attached. A failed plugin can transit to this state if the plugin was in this state or in 'AVAILABLE' state before it failed.

'UNAVAILABLE2' : The plugin is **powered on, initialized, but not identified yet**. Also it may be electrically connected to the related bus, may be it's software-configured, but it's still not available for use by OS and applications because the necessary drivers are not loaded/attached. A plugin without FRUID support may remain in this state before it reaches 'AVAILABLE' state, after that the plugin becomes identified, so it never goes back to this state in its lifetime. A failed plugin can transit to this state if the plugin was in this state before it failed.

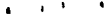
'AVAILABLE' : The plugin is **identified** and **ready for use** by OS and applications, as all the necessary drivers are loaded/attached. The plugin remains in this state **while** in use. A plugin with or without FRUID support is always identified at this state. A failed plugin can transit to this state if the plugin was in this state before it failed.

'FAILED' : The plugin has failed, the reason could be different for different plugins.

'UNKNOWN' : The pluginUnitState value of a plugin should be set to 'UNKNOWN' if COSL fails to find out the actual state(one of the above states) of the plugin.

'No-Plugin' : Though this is not a state a PluginUnit node goes through, COSL will not create any PluginUnit node if there is no plugin installed in a slot or the underlying system fails to determine the presence of a plugin in a slot.

• • • •



la - a powered off and unidentified plugin is removed from the slot
lb - plugin inserted, not powered on, but it's identified
lc - a powered off and identified plugin is removed from the slot

la - a powered off and unidentified plugin is removed from the slot
lb - plugin inserted, not powered on, but it's identified
lc - a powered off and identified plugin is removed from the slot

la - a powered off and unidentified plugin is removed from the slot
lb - plugin inserted, not powered on, but it's identified
lc - a powered off and identified plugin is removed from the slot

2a - an unidentified plugin is removed when it was powered on, and may be it was connected and configured.

be it was connected and configured.

2b - a plugin inserted, powered on, and may be connected & configured

be it was connected and configured.
2b - a plugin inserted, powered on, and may be connected & configured and also the plugin is identified from its FRUID information.

and also the plugin is identified from its FRUID information.

2c - an identified plugin is removed when it was powered on, and may be it was connected and configured.

and also the plugin is identified from its FRUID information.

2c - an identified plugin is removed when it was powered on, and may be it was connected and configured.

3a - an unidentified plugin is powered off

3c - an identified plugin is powered off

the process the plugin is identified through the supported FRUID

through the FRUID information.

- 5 - The plugin is configured or just the drivers are loaded/attached for the h/w resources under an unidentified plugin at Unavailable2 state. At the new state the plugin can be identified because the drivers are attached now. The plugin never goes back to the previous state in its lifetime.
 - 5a - The plugin is configured or just the drivers are loaded/attached for the h/w resources under an identified plugin.
 - 5b - Unconfigure the plugin or unload/detach the drivers.
 - 6 - An unidentified plugin has failed to initialize/ draw power when powered on. 6a - A failed and unidentified plugin is powered off.
 - 6b - An identified plugin has failed to initialize/ draw power when powered on 6c - A failed and identified plugin is powered off.
 - 7 - An unidentified plugin failed at Unavailable2 state. Reason of failure can be different for different plugins, like for CPCI cards it's loss of HEALTHY#.
 - 7a - An unidentified plugin gained its health and it's coming back to Unavailable2 state.
 - 7b - An identified plugin failed at Unavailable1 state. Reason of failure can be different for different plugins, like for CPCI cards it's loss of HEALTHY#.
 - 7c - An identified plugin gained its health and it's coming back to Unavailable1 state.
 - 8 - An identified plugin failed at Available state . Reason of failure can be different for different plugins, like for CPCI cards it's loss of HEALTHY#.
 - 8a - An identified plugin gained its health and it's coming back to Available state.
 - 9 - A plugin when inserted could not power up and initialize itself. This happens when an attempt to make the transitions 2 or 2b fails
 - 9a - A failed plugin is removed from its slot.
 - 10 - A plugin when plugged in its slot, it directly goes to Available state where it's identified and ready-for-use, or in-use. E.g. a fan tray starts blowing as soon as the fan-tray is plugged into a fan slot.
 - 10a - A plugin is removed when it was in Available state, e.g. a fan-tray is removed when it was blowing with proper speed.
- * Transitions to and from UNKNOWN state is possible from and to any of the other pluginUnitStates.

1.3.1.1 Mapping of 'pluginUnitState' to Operational State and Availability Status :

The mapping is as follows -

<i>COSL</i> <i>'pluginUnitState'</i>	<i>MOH</i> <i>OperationalState</i>	<i>MOH</i> <i>AvailabilityStatus</i>	<i>Description</i>
No-Plugin	-	-	
POWEROFF1	"DOWN"	"poweroff"	MOH PluginUnit object is complete with proper identification information, like pluginUnitType, vendorName, version, object name etc. Instead generic values have to be populated.
POWEROFF2	"DOWN"	"poweroff"	MOH PluginUnit object is not complete with proper identification information.
UNAVAILABLE1	"DOWN"	"offline"	MOH PluginUnit object is complete with proper identification information.

<i>COSL</i> <i>'plugInUnitState'</i>	<i>MOH</i> <i>OperationalState</i>	<i>MOH</i> <i>AvailabilityStatus</i>	<i>Description</i>
UNAVAILABLE2	"DOWN"	"offline"	MOH PlugInUnit object is not complete with proper identification information.
AVAILABLE	"UP"	"available"	MOH PlugInUnit object is complete with proper identification information.
FAILED	"DOWN"	"failed"	May or may not complete with proper identification information.
UNKNOWN	"UNKNOWN"	?	May or may not complete with proper identification information.

1.3.1.1.1 Mapping of state transitions and events :

<i>State Transition of</i> <i>1.3.1</i>	<i>COSL Event</i>	<i>MOH Notification</i>	<i>SNMP Trap</i>	<i>Description</i>
1/1b	ems_node_added	AttributeChangeNotification(Slot Status), ObjectCreationNotification	HwUnitChanged(Slot Status), HwUnitCreated	OperationalState = DOWN, AvailabilityStatus = poweroff
2/2b	"	"	"	OperationalState = DOWN, AvailabilityStatus = offline
9	"	"	"	OperationalState = DOWN, AvailabilityStatus = failed
10	"	"	"	OperationalState = UP, AvailabilityStatus = available
1a/1c/2a/2c/9a/10a	ems_node_removed	AttributeChangeNotification(Slot Status), ObjectDeletionNotification	HwUnitChanged(Slot Status), HwUnitDeleted	COSL node deleted, corresponding MOH object deleted.
4a	ems_node_updated	-	-	Update of identification information is done in COSL node, MOH object updates the information as well. OperationalState = DOWN, AvailabilityStatus = poweroff
4	"	AttributeChangeNotification(Availability Status)	-	Update of identification information is done in COSL node, MOH object updates the information as well. OperationalState = DOWN, AvailabilityStatus = offline
3/3b	"	"	-	OperationalState = DOWN, AvailabilityStatus = offline
3a/3c	"	"	-	OperationalState = DOWN, AvailabilityStatus = poweroff
5	"	StateChangeNotification, AttributeChangeNotification(Availability Status)	HwUnitUp	Update of identification information is done in COSL node, MOH object updates the information as well. OperationalState = UP, AvailabilityStatus = available
5a	"	"	"	OperationalState = UP, AvailabilityStatus = available
5b	"	"	HwUnitDown	OperationalState = DOWN, AvailabilityStatus = offline
6/6b/7/7b	"	AttributeChangeNotification(Availability Status)	-	OperationalState = DOWN, AvailabilityStatus = failed

<i>State Transition of 1.3.1</i>	<i>COSL Event</i>	<i>MOH Notification</i>	<i>SNMP Trap</i>	<i>Description</i>
8	"	StateChangeNotification, AttributeChangeNotification(Availability Status)	HwUnitDown	OperationalState = DOWN, AvailabilityStatus = failed
6a/6c	"	AttributeChangeNotification(Availability Status)	-	OperationalState = DOWN, AvailabilityStatus = poweroff
7a/7c	"	"	-	OperationalState = DOWN, AvailabilityStatus = offline
8a	"	StateChangeNotification, AttributeChangeNotification(Availability Status)	HwUnitUp	OperationalState = UP, AvailabilityStatus = available
to UNKNOWN	"	StateChangeNotification	-	OperationalState = UNKNOWN, AvailabilityStatus = ?
from UNKNOWN	"	"	-	OperationalState, AvailabilityStatus values depend on the new COSL plugInUnitState from UNKNOWN.

1.3.2 Mapping of hardware/OS states of the CPCI cards to the 'pluginUnitState' :

This mapping is exclusively a COSL issue. At first the hardware/OS states are explained and then the mapping is presented in a table.

The following states are defined in accordance to the PICMG hot-swap specification and the Solaris for Montecarlo/Tonga :

'P1' : Plugin is just present in the slot, but it's **not powered on**, not initialized, not active on PCI bus.

'H1' : Plugin is **powered up**, initialized, but **not connected** electrically to PCI bus, so not active on PCI bus.

'H2' : Plugin is electrically **connected** to the PCI bus, so active on PCI bus.

'S1' : Plugin is **configured**, i.e. the software senses the plugin and allocates PCI resources for the plugin, BUT the necessary **drivers** for the h/w resources under the plugin are **not loaded/not associated**.

'S2' : Plugin is **configured** as well as the necessary **drivers are loaded/associated**. So the resources under the plugin are **ready to be used** by the OS or applications.

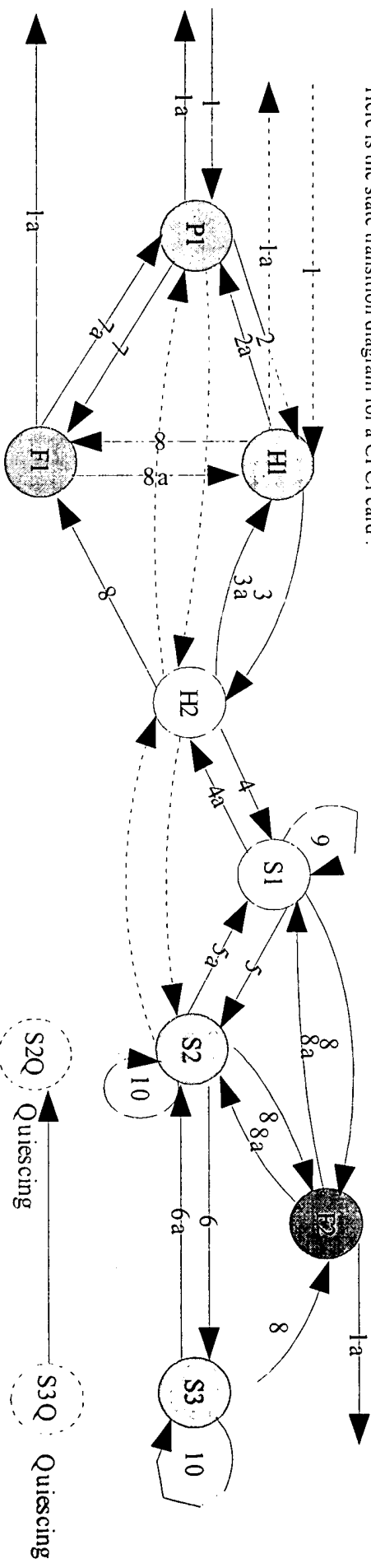
'S3' : The plugin is **in use** by OS or applications.

'F1' : This is one of the two **failed** states. This is 'H1F' in PICMG. At this state the plugin is disconnected from the PCI bus, though it remains powered on.

Basically, at this state the plugin is not in suitable condition to be connected to the PCI bus.

'F2' : This is the other **failed** state which is not defined in PICMG, but Solaris implements this state. This state is reached whenever there is a loss of HEALTHY# at S1, S2, or S3. And the plugin can come back to S1, or S2 on a gain of HEALTHY#. Though the plugin is **disconnected** from the PCI by the h/w layer, it remains **configured**.

Here is the state transition diagram for a CPCI card :



The transitions with dotted lines are automatic transitions.

1 - board physically inserted into the slot,

2 - power on

3 - connect to PCI bus

4 - configure

1a - board removed from the slot

2a - power off

3a - disconnect from PCI bus

4a - unconfigure

5 - load/attach necessary drivers

6 - OS or applications start using the board

5a - unload/detach the drivers

6a - the last user is done, no more user of the board

7 - the board has failed to initialize when powered on, sets loss of HEALTHY#.

7a - A failed CPCI card is powered off (HA hotswap capability)

8 - Loss of HEALTHY#

9 - Drivers not installed, or loading of drivers failed

Mapping...

On identification of the CPCL card - The CPCL cards with FRUID support are identifiable even before the necessary drivers are loaded/attached. Otherwise they can be identified only when they reach S2 state for the first time, then after they remain identified until they are taken out from the system. Depending on the fact that a CPCL card has been yet identified or not , the pluginUnitState for the card at COSL will vary. So the pluginUnitState of a pci card is determined by the state of the card in the above state machine as well as by the fact whether the card has been identified or not.

At MOH, the PluginUnit object corresponding to the card will populate proper identification information like pluginUnitType, vendorName, version etc when the card gets identified, otherwise it'll be shown as an 'unknown' PluginUnit object.

Now, lets see how the above states map to 'pluginUnitState' as described in 1.3.1.

<i>Hardware/OS State of CPCL card</i>	<i>COSL 'pluginUnitState'</i>	<i>MOH OperationalState</i>	<i>MOH AvailabilityStatus</i>	<i>Description</i>
P1, Identified	POWEROFF1	"DOWN"	"poweroff"	This state is reachable by a card supporting HA-hotswap capability. A card without FRUID support can reach this state only after it has reached S2 at least once in its lifetime. MOH PluginUnit object is complete with proper identification information.
P1, Unidentified	POWEROFF2	"DOWN"	"poweroff"	This state is reachable by a card supporting HA-hotswap capability. MOH PluginUnit object is 'unknown' yet.
H1/H2/S1, Identified	UNAVAILABLE1	"DOWN"	"offline"	MOH PluginUnit object is complete with proper identification information.
H1/H2/S1, Unidentified	UNAVAILABLE2	"DOWN"	"offline"	MOH PluginUnit object is 'unknown' yet.
S2/S3, (at these states the pci cards are always identified)	AVAILABLE	"UP"	"available"	MOH PluginUnit object is complete with proper identification information.
F1/F2	FAILED	"DOWN"	"failed"	The card has failed. MOH PluginUnit object can be either an 'unknown' plugin or an identified plugin.
-	UNKNOWN	"UNKNOWN"	?	COSL is unable to find out the exact state of the pci card.

1.3.2.1 Mapping of state transitions and events :

<i>State Transition of 1.3.2</i>	<i>'PluginUnitState' Transition(1.3.1)</i>	<i>COSL Event</i>	<i>Description</i>
1(to H1) / 1(to H1),3	2	ems_node_added	A CPCL card with no FRUID support has been inserted. Since the card doesn't support FRUID, the card can't be identified yet, COSL should populate only generic/no information about the card type, vendor, version in the corresponding node. The 'pluginUnitState' value should be 'UNAVAILABLE2'. No child nodes yet.
"	2b	ems_node_added	A CPCL card with FRUID support has been inserted. The 'pluginUnitState' value should be 'UNAVAILABLE1' as the card is identified from the FRUID information. COSL should also create the children nodes under this plugin.
3,4,5 / 4,5 / 5	5	ems_node_updated	The card gets identified and usable; the new 'pluginUnitState' value should be 'AVAILABLE'. The card will never return to UNAVAILABLE2 in its lifetime. But if driver loading fails the card should remain in the previous state only, and COSL should not generate any event in this situation. The subtree under the card should also get created in COSL.

<i>State Transition of 1.3.2</i>	<i>'PlugInUnitState' Transition(1.3.1)</i>	<i>COSL Event</i>	<i>Description</i>
"	5a	ems_node_updated	An identified card becomes usable; the new 'plugInUnitState' value should be 'AVAILABLE'. But if driver loading fails the card should remain in the previous state only, and COSL should not generate any event in this situation.
5a / 5a,4a / 5a,4a,3a	5b	ems_node_updated	to UNAVAILABLE1
1,7	9	ems_node_added	A CPCI card failed to power up and initialize just after it was inserted. A PlugInUnit node should be created for the inserted card in COSL, the 'plugInUnitState' value should be 'FAILED'. This happens in case of basic and full hotswap.
8 (from H1/H2/S1/S2/S3)	7 / 7b / 8	ems_node_updated	to FAILED
8a (to H1/S1)	7a	ems_node_updated	to UNAVAILABLE2 for an unidentified card
"	7c	ems_node_updated	to UNAVAILABLE1 for an identified card
8a (to S2)	8a	ems_node_updated	to AVAILABLE
1a (from H1)	2a/2c	ems_node_removed	The subtree starting from the node for the CPCI card has to be deleted, in COSL.
1a (from F1/F2)	9a	ems_node_removed	same as above

The following transitions are valid for full hotswap only :

<i>State Transition of 1.3.2</i>	<i>'PlugInUnitState' Transition(1.3.1)</i>	<i>COSL Event</i>	<i>Description</i>
1(to H1),3,4,5	10 (actually 2, 5)	ems_node_added	A PlugInUnit node should be created for the inserted card in COSL, the 'plugInUnitState' value should be 'AVAILABLE'. Since the card is identified, COSL should create all the nodes under the card.
"	10 (actually 2b, 5a)	"	Same as above
1(to H1),3,4,9	2	"	A PlugInUnit node should be created for the inserted card in COSL, the 'plugInUnitState' value should be 'UNAVAILABLE2'.
"	2b	"	to 'UNAVAILABLE1' for an identified card. This behaviour can be shown by a card with FRUID.
5a,4a,3a,1	10a (actually 5b, 2c)	ems_node_removed	The subtree starting from the node for the CPCI card has to be deleted, in COSL.

The following transitions are valid for HA hotswap only :

<i>State Transition of 1.3.2</i>	<i>'PlugInUnitState' Transition(1.3.1)</i>	<i>COSL Event</i>	<i>Description</i>
1(to P1)	1	ems_node_added	A PlugInUnit node is created with the 'plugInUnitState' value set to 'POWEROFF2'. No child nodes yet.
1(to P1)	1b	ems_node_added	A card with FRUID and HA hotswap capability is plugged in. The 'plugInUnitState' value set to 'POWEROFF1'.

State Transition of 1.3.2	'PluginUnitState' Transition(1.3.1)	COSL Event	Description
2 / 2.3 / 2.3.4	4	ems_node_updated	card is identified. Children nodes are to be created. The card has FRUID support.
-	4a	ems_node_updated	card is identified. Children nodes are to be created. No MOH notification. The card has FRUID support.
2 / 2.3 / 2.3.4	3	ems_node_updated	
"	3b	ems_node_updated	The card has FRUID support.
2a / 3a,2a / 4a,3a,2a	3a	ems_node_updated	
"	3c	ems_node_updated	The card has FRUID support.
7	6	ems_node_updated	
"	6b	ems_node_updated	The card has FRUID support.
7a	6a	ems_node_updated	
"	6c	ems_node_updated	The card has FRUID support.
1a	1a/1c	ems_node_removed	The subtree starting from the node for the CPCI card has to be deleted, in COSL.

1.3.2.2 The effect of 'pluginUnitState' of a CPCI card node on its children nodes

Creation of the subtree under the plugin : Whenever a CPCI card makes transition into one of the following three states, POWEROFF1, UNAVAILABLE1 and AVAILABLE, either just after the plugin is created or from one of the following two states, POWEROFF2, UNAVAILABLE2, the child subtree is created. In other words whenever the CPCI card gets identified(this happens only once in its lifetime, after that it remains identified), the subtree under the card has to be created.

If the card supports FRUID information, the subtree can be created as soon as the card transits to POWEROFF1 state, if it doesn't, the creation of the subtree under the plugin has to wait till the plugin reaches the AVAILABLE state. MOH, upon receiving an ems_node_updated event from COSL for a change in 'pluginUnitState' for a plugin which was unidentified till now, should get the subtree from the COSL and then should create the equivalent subtree in its hierarchy, if the new 'pluginUnitState' is one of the identified states.

Deletion of the subtree under the plugin : When an identified CPCI card is removed from the its slot, the subtree starting from the plugin node has to be deleted. MOH will delete the equivalent subtree starting from the plugin object for this CPCI card from its own hierarchy.